CLAIMS

We claim:

1. A debug interface, comprising:

logic responsive to a pre-fork event, the pre-fork event responsive to a fork instruction call wherein the pre-fork event includes indicia that identifies a child process to be created in accordance with the fork instruction call.

2. The interface of claim 1, wherein the indicia comprises a process identifier.

3. The interface of claim 1, wherein the pre-fork event is delivered by a parent process to a software monitor.

4. The interface of claim 3, wherein the parent process was generated by the software monitor.

5. The interface of claim 3, wherein the parent process was instrumented for run-time analysis by the software monitor.

6. The interface of claim 3, wherein the pre-fork event is delivered before the fork instruction call is executed by the parent process.

7. The interface of claim 3, wherein the software monitor responds by executing the child process until completion.

8. The interface of claim 7, wherein the software monitor responds to indicia of completion of the child process by executing the parent process until completion.

9. The interface of claim 8, wherein the software monitor ensures that the first event in the parent process that spawned the child process is the fork event itself.

10. A method for controlling the execution of a child process created from a parent process, wherein the parent process is instrumented by a software tool, the method comprising the steps of:

receiving indicia that a fork instruction will be executed by the parent process;

suspending execution of the parent process;

extracting a process identifier from the indicia of the fork instruction, the process identifier corresponding to a child process to be generated by the parent process when the parent process executes the fork instruction;

setting a process monitor thread to observe the child process; and

resuming execution of the parent process to enable the parent process to execute the fork instruction.

11. The method of claim 10, further comprising:

waiting for indicia that the child process has nominally terminated; and

setting a process monitor thread to observe the parent process.

12. The method of claim 11, wherein setting a process monitor thread comprises enabling observation of trace events generated by the parent process.

13. The method of claim 10, wherein receiving indicia comprises a pre-fork event.

14. The method of claim 13, wherein the pre-fork event includes the process identifier.

15. The method of claim 10, wherein setting a process monitor thread comprises enabling observation of trace events generated by the child process.

16. A method for executing a parent process instrumented by a software tool to ensure execution of a child process when the parent process contains a fork instruction, the method comprising the steps of:

determining if a fork instruction is about to be executed;

generating a pre-fork event that includes indicia of a child process that will be generated by the fork instruction;

sending the pre-fork event to the software tool;

waiting for indicia that the software tool successfully processed the pre-fork event;

executing the fork instruction; and

suspending execution of the parent process.

17. The method of claim 16, further comprising:

waiting for indicia that the child process has terminated; and

resuming execution of the parent process.

18. The method of claim 17, wherein waiting for indicia that the child process has terminated comprises a trace event.

19. The method of claim 16, wherein indicia of the child process comprises a process identifier.

20. A method for controllably switching a target process of a process monitor thread between an instrumented parent process and a child process generated by the parent process, the method comprising the steps of:

determining if initiation of a child process was successful;

when the initiation of the child process is unsuccessful, waiting a predetermined amount of time before checking if a parent process responsible for creating the child process received indicia of a failure of a fork instruction that created the child process ; and

setting indicia of a process to monitor to the parent process; otherwise,

checking if a parent process responsible for creating the child process received indicia of a failure of the fork instruction that created the child process;

when execution of the fork instruction was unsuccessful, notifying a software monitor of the failed condition; otherwise,

monitoring the designated target process.

21. The method of claim 20, wherein determining if initiation of a child process was successful comprises checking the status of a debugging instruction.

22. The method of claim 20, wherein checking if a parent process responsible for creating the child process received indicia of a failure comprises searching for a trace event while performing a non-blocking trace wait on the parent process.

23. The method of claim 20, further comprising:

aborting child process monitoring when the initiation of the child process is unsuccessful.

24. The method of claim 20, wherein notifying a software monitor of the failed condition comprises an indication that the parent process started two processes with the same process identifier.

25. An operating system, comprising:

a pre-fork event, the pre-fork event responsive to a fork instruction wherein the pre-fork event includes indicia that identifies a child process to be created in accordance with the fork instruction.

26. The operating system of claim 25, wherein the indicia comprises a process identifier.

27. A computer readable medium, comprising:

logic responsive to a pre-fork event, the pre-fork event responsive to a fork instruction wherein the pre-fork event includes indicia that identifies a child process to be created in accordance with the fork instruction.

28. The computer readable medium of claim 27, wherein the indicia comprises a process identifier.